**SQL**

**To check SQL version (cmd)-**

- Select version();

**Five SQL Commands-**

1. DDL (Data Defination Language) –

- create

- alter

- drop

- truncate

2. DML (Data Manipulation Language) –

- insert

- update

- delete

3. DQL (Data Query Language) –

- select

4. DCL (Data Control Language) –

- grant

- revoke

5. TCL (Transaction Control Language) –

- Commit

- Roll back

- Save point

**SQL Queries –**

**Database queries-**

create database db\_name; (to create new database folder)

drop database db\_name; (to delete database)

use database db\_name; (to use your database to store the created table)

**Table queries-**

create table table\_name; (rollno int, name varchar(20),contact long); (to create new table)

drop table table\_name(); (to delete table)

select \* from table\_name; (to view full created table)

select column\_name from table\_name; (to view specific column)

insert into table\_name values; (101,rohan,8669997070); (to insert values in that table)

#to use table from this database

use expo;

**Example-**

**#create table**

create table employee(empid int,name varchar(20),address varchar(50),contact long,mail varchar(20),department varchar(15));

**#single insertion of value into the table**

insert into employee values(101,'rohan','ravet',8669997070,'rohanpote10@gmail.com','development');

**#partially insertion of value into the table**

insert into employee(empid) values(110); 🡪(Only inserts empid other values will take default (null))

**#display total table**

select \* from employee;

**#multiple insertion of values into the table**

insert into employee values

(102,'aniket','dehuroad',7249852858,'aniketnangare23@gmail.com','development'),

(103,'yash','dehuroad',9680754321,'yashkumbhae03@yahoo.com','testing'),

(104,'shrikant','ravet',8907654432,'shreepoul11@gmail.com','development'),

(105,'sarthak','punawale',8897654321,'sarthak123@reddit.com','hr'),

(106,'adarsh','bhuaon',9087654321,'adarsh987@yahoo.com','testing'),

(107,'vikrant','ravet',7890654321,'vikrant908@gmail.com','ticketsolving'),

(108,'nachiket','pcmc',8889097654,'nachiket667@gmail.com','hr');

**#to display specific column**

select name from employee;

**#display specific row of the table**

select \* from employee where name='yash';

**#** **drop table**

Drop table table\_name;

Drop column

**#comparison**

select \* from employee where empid>105;

select \* from employee where empid>=105;

select \* from employee where empid<107;

select \* from employee where empid<=107;

select \* from employee where empid!=101;

select \* from employee where empid<>101;

**#logical operator**

**#and (checks the row not all table)**

select \* from employee where empid=101 and address='ravet';

select \* from employee where name='yash' and address='punawale';

select \* from employee where empid=104 and name='shrikant' and address='ravet';

**#or (checks whole table)**

select \* from employee where empid=102 or name='shrikant';

select \* from employee where name='aniket' or address='dehuroad' or empid=100;

select \* from employee where empid=104 or empid=107;

**#not (it will show the all values except that NOT condition)**

select \* from employee where not empid=105;

select \* from employee where not address='ravet';

select \* from employee where not name='sarthak';

select \* from employee where not empid=101 and not name='shrikant';

**#between (shows in between values (only numbers))**

select \* from employee where empid between 104 and 106;

select \* from employee where empid not between 104 and 106;

**#in (values entered in ()are shown)**

select \* from employee where empid in(102,105,107);

select \* from employee where name in('yash','vikrant','shrikant');

select \* from employee where empid=102 or empid=104 or empid=107;

**#like (with help of like 'starting of word, ending of word, inbetween word' we can find the row)**

select \* from employee where name like'%ket'; #(enter the end of the word you want to find it will show that rows)

select \* from employee where name like'ani%'; #(enter the start of the word you want to find it will show that rows)

select \* from employee where address like'%ve%'; #(enter the middle of the word you want to find it will show that rows)

**#update -->(address will get updated where empid is 103)**

update employee set address='ravet' where empid=103;

update employee set name='yash kumbhar' where empid=103;

update employee set name='rohan',address='pcmc',contact=9988776655 where empid=101; 🡪(to update whole row)

**#delete -->(delete specific row or rows)**

delete from employee where empid=109;

**#alter -->(it will add column, drop column)**

alter table employee add age int;

alter table employee drop column age;

alter table employee add salary double;

alter table employee modify salary decimal(9,3);

update employee set salary=200000.500 where empid=101;

update employee set salary=100000.101 where empid=102;

**#Create table - to store in database by using path of file**

create table filedemo(id int, file1 blob); 🡪(blob 🡪path of file)

desc filedemo;

insert into filedemo values(101,load\_file('C:\\Users\\Rohan\\Desktop\\DemoGit\\ExpoRepository'));

🡪(enter your files path which yoy want to store in db)

select \* from filedemo;

create table flagtable(id int, flag boolean); 🡪(for boolean values)

desc flagtable;

insert into flagtable values(113,false); 🡪(true=1 & fale=0)

select \* from flagtable;

**#Date**

create table datedemo

(id int, registerdate date,dob date,time1 time,dt1 datetime,year1 year);

desc datedemo;

insert into datedemo values

(5001,curdate(),'2000/04/10',curtime(),now(),'2024');

select \* from datedemo;

insert into datedemo values

(5002,curdate(),'2000/09/26','12:45:15','2006/05/14 07:30:27','2015');

delete from datedemo where id = 5002;

* **Constraints:**

create table expo(id int not null, name varchar(20)not null);

🡪(it doesn’t show the default null values)

select \* from expo;

insert into expo values(101,'rohan');

insert into expo(id) values(102); 🡪(it will show blank in name colum )

**#UNIQUE**

create table student(rollno int unique,name varchar(20),pancardno varchar(20) unique);

🡪(in front of datatype we can add unique 🡪 we cannot add same field multiple time)

desc student;

insert into student values(2,'rohan','RTYEW1234E');

SELECT \* FROM student s;

insert into student values(3,'YASH','RTYEW1234E'); 🡪(shows error🡪can’t add same fields)

insert into student(rollno,name) values(4,'ruchika');

insert into student(name,pancardno) values('ganesh','FGHJK6676L');

**#Primary Key 🡪**

**- we cannot take null values.**

**- 1 primary key for 1 table column**

create table employee1(eid int primary key,name varchar(20) not null);

desc employee1;

insert into employee1 values(111,'rohan');

insert into employee1 values(112,'yash');

SELECT \* FROM employee1 e;

insert into employee1 values(113,'aniket');

insert into employee1(name) values('sarthak');

**- or we ccan use not null & unique key also**

**- we can take null values.(Unique keyword)**

**- multiple unique key for 1 table column**

create table employee2(eid int not null unique,ename varchar(20),

pancard varchar(20) unique,aadharno varchar(20) not null unique);

desc employee2;

insert into employee2 values(101,'Aakash','WQWER1234Q',789456123301),(102,'Abhijeet','WQUYD1234Q',787894523301);

select \* from employee2;

**#auto increment- (increments 1 by 1, works with primary key only)**

create table student1(rollno int primary key auto\_increment,name varchar(20));

insert into student1(name) values('sarthak');

select \* from student1;

insert into student1(name) values('yash');

insert into student1 values(51,'rohan');

insert into student1(name) values('dayanand');

**#Foreign key-**

create table department(did int primary key, dname varchar(20));

create table employee3(eid int primary key,ename varchar(20), salary int,

deptid int, FOREIGN KEY(deptid) references department(did));

desc employee3;

insert into department values(101,'HR'),(102,'IT'),(103,'Admin'),(104,'Finance');

select \* from department;

insert into employee3 values(201,'sarthak',35000,101),(202,'aatharva',30000,104),(203,'shridhar',80000,102);

select \* from employee3;

insert into employee3(eid,ename,salary) values(204,'sanket',60000);

**#Default- (it will take provided default values)**

use expo;

create table studentDefault

(id int ,name varchar(20),deptname varchar(20) default 'Unknown');

insert into studentDefault values(110,'raj','Mechanical');

insert into studentDefault values(114,'jay','IT');

SELECT \* FROM studentDefault s;

insert into studentDefault(id,name) values(121,'ganesh');

**#Check- (it will only works on workbench)**

create table personCheck(id int ,name varchar(20),age int check(age>=18));

insert into personCheck values(111,'rohan',24),(112,'yash',25),(113,'aniket',17);

select \* from personCheck;

**#delete table- (it will delete whole table if condition is not given )**

* **If auto\_increment is used & we deleted the table & again added new values it will auto\_increment from that deleted row(it didn’t reset counter)**

delete table Student;

delete from Student where id = 101; 🡪

(if condition given then only row or rows will ge deleted)

**#truncate table- (it will delete whole table if condition is not given )**

* **If auto\_increment is used & we deleted the table & again added new values then it will auto\_increment from start (it resets the counter ).**

truncate from Student;

* **Joins-**

use expo;

create table customer(cid int primary key,cname varchar(20));

insert into customer values(101,'Rohan'),(102,'Aniket'),(103,'Yash'),(104,'Sarthak');

insert into customer values(105,'aakash');

select \* from customer;

create table product(pid int primary key,pname varchar(20),price int,id int,foreign key(id) references customer(cid));

insert into product values(201,'laptop',60000,102),(202,'tv',50000,103),(203,'fridge',40000,104),

(204,'mobile',80000,101),(205,'microwave',25000,103);

insert into product(pid,pname,price) values(206,'book',750);

insert into product(pid,pname,price) values(207,'trimmer',2000);

select \* from product;

**#inner join- (it will display common values from 2 tables by using foreign key)**

**-(or we can add without PK & FK but values of both tables specifying column should be match )**

select customer.cname, product.pname,product.price

from customer inner join product

on customer.cid = product.id;

**#left outer join- (it display left table whole data & both tables matching data)**

select customer.cname, product.pname,product.price

from customer left join product

on customer.cid = product.id;

**#Right Outer join-(it display right table whole data & both tables matching data)**

select customer.cname, product.pname,product.price

from customer right outer join product

on customer.cid = product.id;

**#Full Outer Join- (combines whole table)(but not works in SQL & workbench)**

select customer.cname, product.pname,product.price

from customer full outer join product

on customer.cid = product.id;

**#Cross Join-(1st table every single row joins with 2nd tables whole rows)**

create table cars(carid int,carname varchar(20));

create table colours(colourid int,colourname varchar(20));

insert into cars values(1,'Xuv 700'),(2,'Thar'),(3,'Creta'),(4,'Virtus');

insert into colours values(1,'Black'),(2,'Red'),(3,'White');

SELECT \* FROM cars c;

SELECT \* FROM colours c;

select cars.carname, colours.colourname from

cars CROSS JOIN colours;

**#Union & Union** **All -**

create table batch1(id int,name varchar(20));

create table batch2(id int,name varchar(20));

insert into batch1 values(1,'yash'),(2,'aniket'),(3,'rohan');

insert into batch2 values(1,'yash'),(5,'raj'),(6,'sanket'),(7,'aadiitya');

SELECT \* FROM batch1 b;

SELECT \* FROM batch2 b;

**-Union – (Combine both tables whole except duplicate values)**

**(must have same datatypes & order in both tables)**

SELECT id,name FROM batch1

UNION

SELECT id,name FROM batch2;

**-Union All – (Combine both tables whole values)**

**(must have same datatypes & order in both tables)**

SELECT id,name FROM batch1

UNION All

SELECT id,name FROM batch2;

**#Order By clause-**

select \* from product;

select \* from product

order by pname ASC; 🡪(display column in ascending order)

select \* from product

order by pname desc; 🡪(display column in descending order)

select \* from product

order by price desc;

select price from product order by price asc;

🡪(only display single column in ascending order)

**#Distinct-**

select distinct price from product; 🡪(displays column values except duplicates)

**#Aggregate functions-**

select max(price) from product; 🡪(displays max value from that column)

select min(price) from product; 🡪(displays min value from that column)

select sum(price) from product; 🡪(displays sum value of that column)

select avg(price) from product; 🡪(displays avg value of that column)

select count(pname),count(distinct price) from product;

🡪(displays total count of that column)

**#Alias-**

select max(price) as Largestprice from product;

🡪(displays max value from that column with that given name)

select min(price) as lowestprice from product;

🡪(displays min value from that column with that given values)

use test;

create table employee(eid int,ename varchar(20),dname varchar(20));

insert into employee values(1,'rohan','IT'),(2,'yash','admin'),(3,'aniket','hr'),(4,'sarthak','IT'),(5,'aakash','admin'),

(6,'abhijeet','IT'),(7,'ashok','admin'),(8,'shri','IT');

SELECT \* FROM employee e;

**#Count –** (it displays count of similar data in column)

select count(ename),dname from employee

group by dname;

🡪(it displays total employees count from same department)

select count(eid),dname from employee

where dname='IT' group by dname; 🡪(it display specific column content count)

select count(eid),dname from employee

group by dname having dname='hr' ; 🡪(it display specific column content count)

select count(eid),dname from employee

group by dname order by dname desc ;

🡪(it displays total employees count from same department in descending order)

**#Finding nth largest price-**

use expo;

SELECT \* FROM expo.product p;

select price from product;

select distinct price from product;

select max(distinct price) as largestprice from product;

select price from product

where price< (select max(distinct price) as largestprice from product);

🡪(it displays all values except largest one)

select max(price) from product

where price< (select max(distinct price) as largestprice from product);

🡪(it displays 2nd largest price)

select max(price) as 2ndlargestprice from product

where price< (select max(price) from product

where price< (select max(distinct price) as largestprice from product));

🡪(it displays 3rd largest price)

**#LIMIT-** (it displays upper values as per given limit)

select distinct price from product;

select distinct price from product order by price desc;

select distinct price from product order by price desc limit 2; 🡪(it displays largest 2 price)

select min(price) from (select distinct price from product order by price desc limit 2) as product1; 🡪(it displays min value from descending limit 2)

**#LIMIT & Offset –** (offset skips the rows we want to enter & displays the rest of table)

select distinct price from product order by price desc limit 1 offset 1;

🡪(it displays 2nd largest price)

**#Grant& Revoke-**

Grant🡪gives access to user

Revoke🡪revokes access from user

SELECT \* FROM expo.employee3 e;

use expo;

create user 'user1'@'localhost' identified by 'user1';

show grants for 'root'@'localhost';

show grants for 'user1'@'localhost';

grant select on expo.employee3 to 'user1'@'localhost';

grant select,insert on expo.employee3 to 'user1'@'localhost';

revoke select on expo.employee3 from 'user1'@'localhost';

**#autocommit,commit –**

SELECT \* FROM employee3 e;

insert into employee3 values(205,'ameya',20000,103);

set autocommit=0; 🡪(it autocommits false value (onwards values are stored temp))

insert into employee3 values(207,'zaid',40000,103);

commit; 🡪(onwards stored values are permanant)

insert into employee3 values(208,'yamini',30000,102),(209,'nehal',50000,101);

savepoint sp1; 🡪(it will save till now data)

delete from employee3;

SELECT \* FROM employee3 e;

rollback to sp1; 🡪(it will restore that savepoint data)

**#Procedure-** (right click on your table & select procedure option)

DELIMITER $$

CREATE PROCEDURE `expo`.`employeeDetails` () 🡪(reference name)

BEGIN

select \* from employee3 e;

🡪(we added display table query here now we can display table by using reference name of this procedure)

🡪(we can add any type of query here like insert,update, delete like that)

END $$

DELIMITER ;

call employeeDetails(); 🡪(we can call that procedure by using there name)

**#Functions –** (We can add function in this like addition,substraction likr that)

SELECT \* FROM customer c;

SELECT \* FROM product p;

DELIMITER $$

CREATE FUNCTION `expo`.`calculateddiscount` (price int) RETURNS INT

BEGIN

declare discountedprice int;

set discountedprice = price-(0.1\*price);

return discountedprice;

END $$

DELIMITER ;

select min(price) from product;

select calculateddiscount(price) from product;

select pname,price as originalprice, calculateddiscount(price) as discountedprice from product; 🡪(it will display the product table pname,price & the discount applied column)

**#View-** (it will display whatever you want to display to customer)

SELECT \* FROM employee3 e;

create view employee\_view as select ename,salary from employee3;

select \* from employee\_view;

SELECT \* FROM customer c;

SELECT \* FROM product p;

create view customer\_product as

select pid,pname,cname from

customer inner join product

on customer.cid = product.id; 🡪(it will display the data from 2 tables by using join)

SELECT \* FROM customer\_product c;

drop view employee\_view;

**#Index-** (so we can easily access that particular name by using index)

show index from employee3;

create index ename\_index on employee3(ename);

**#Coalesce**-

(it is useful when we create index then null values cannot index so by using **coalesce** we can give a name to all null values)

SELECT \* FROM person3 p;

desc person3;

insert into person3(id) values(450);

select id, coalesce(name,'unknown') from person3;

🡪(all nul values will be replace by Unknown)